Introduction to Geographically Weighted Regression

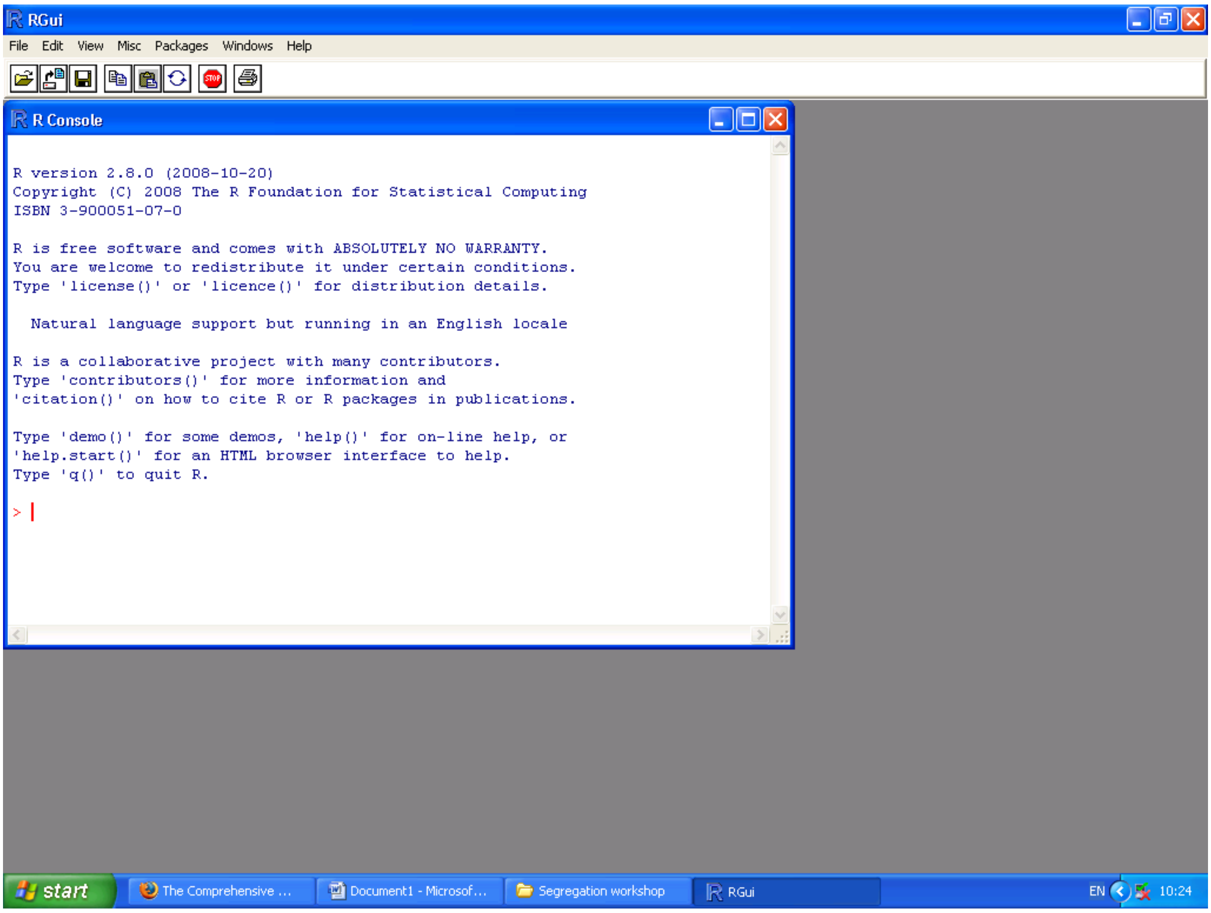
Outline

This practical session is intended as a beginner's introduction to Geographically Weighed Regression (GWR). It is by no means comprehensive. For much more detail and a better understanding of the statistical foundations of GWR please see Fotheringham *et al.* (2002) Geographically Weighted Regression: The Analysis of Spatially Varying Relationship, published by Wiley.

There is "standalone" GWR software that you can obtain from http://ncg.nuim.ie/ncg/GWR/. We, however, shall be using the version available in R, free statistical and computing software that you can obtain from http://cran.r-project.org/. The discussion of R and how it works is here kept to a minimum. See the previously referenced website if you would like to learn more.

Getting going

Locate the R program in the Windows Start menu and run it. This will load the "R Gui" (so called, but use of R is primarily command line).



```
R version 2.8.0 (2008-10-20)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Figure 1. Screen shot of the "R Gui" as the program starts

Locate the comma delimited file, “southeast.csv” and, within the “R Console” type (after the > symbol) an instruction of the form:

```
mydata = read.csv(“H://segregation/southeast.csv”, header=TRUE)

# Anything that appears after # is a comment and doesn’t need to
# be typed. The above code reads the data into an R data frame
# called “my data”
```

Note that R is case sensitive and also that if you make a mistake you can scroll up through your recently typed commands using the ↑ key on your keyboard.

To inspect the top of the data frame, type

```
head(mydata)
```

To see the variable names

```
names(mydata)
```

To make those variables ‘usable’ in R type

```
attach(mydata)
```

The data frame contains Census information collected for 2536 zones in South East England. For each of those a grid reference marking the population weighted centre of the zone is given. We can produce maps from this information, for example of how the proportion of households not owning a car (or van) varies across the country.

```
library(sp)

# Loads the ‘add-ins’ for R that are used for mapping

map = SpatialPointsDataFrame(data=mydata,  coords=cbind(POPEAST,
POPNORTH))

# Makes the data into a “mappable” format

names(map)

# Lists the “mappable” data

colours = c(“dark blue”, “blue”, “red”, “dark red”)

# Creates a colour palette

spplot(map, “Nocars”, cuts=quantile(Nocars), col.regions=colours)

# To provide more detail to the map produce above, use:

spplot(map, “Nocars”, cuts=quantile(Nocars), col.regions=colours,
cex=0.3)
```

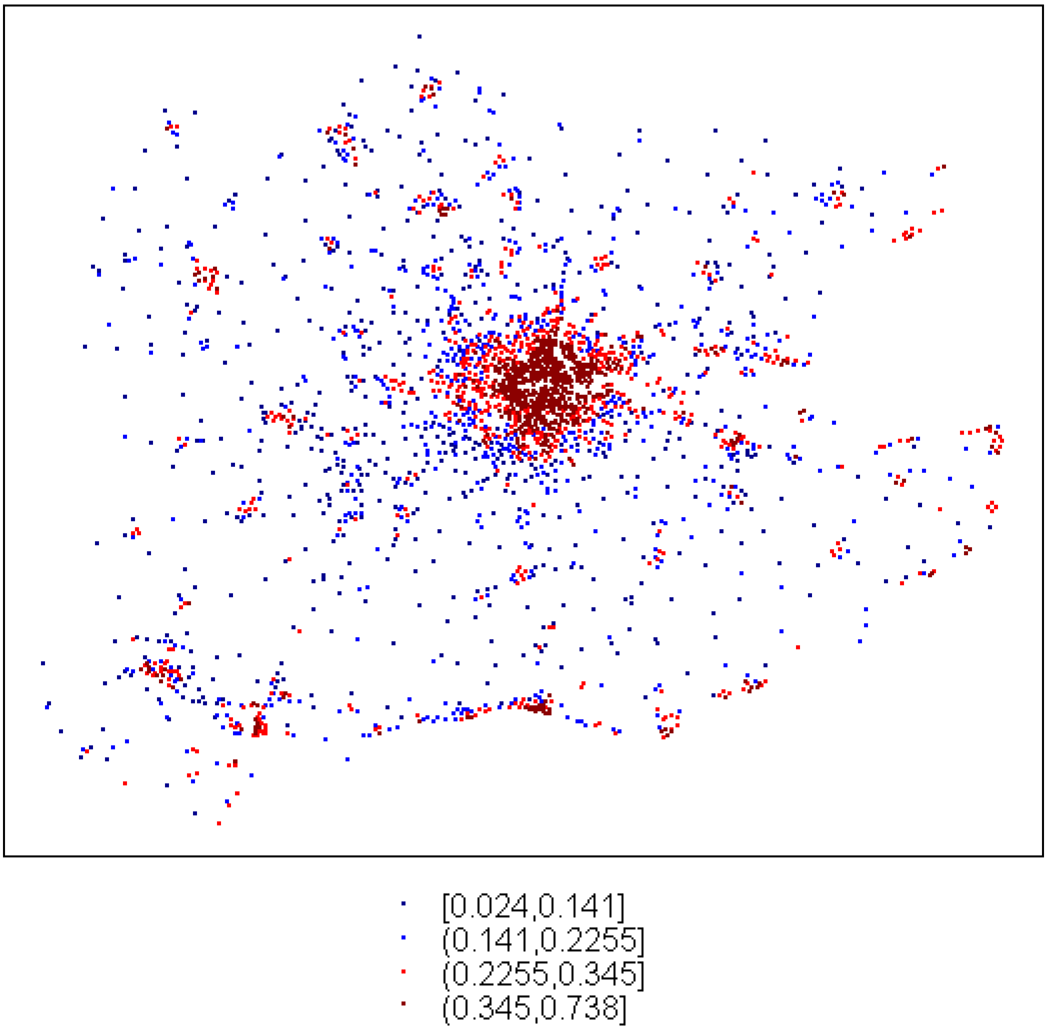


Figure 2. Map showing geographical variation in car non ownership in the SE of England. The map is shaded by quartiles.

We will now fit a simple, linear regression model (a line of best fit) to explore the relationship between the proportion of households without a car (Y) and the proportion of economically active persons unemployed (X) in each Census zone.

First, we produce a plot of the relationship to check it is broadly linear:

```
plot(Nocars ~ Unemp, data=map)
```

Then we fit the regression line:

```
model1 = lm(Nocars ~ Unemp, data=map)

# Fits the linear model, hence lm(…)

abline(model1, col=”red”)
```

```
# Draws the regression line on the plot
```

```
summary(model1)
```

Not unsurprisingly, the relationship is significant at a very high confidence level. But, what about the residuals – is there a geographical patterning to where the model over- or under-predicts? Let’s look!

```
resids = residuals(model1)
```

```
colours = c(“dark blue”, “blue”, “red”, “dark red”)
```

```
map.resids = SpatialPointsDataFrame(data=data.frame(resids),
coords=cbind(POPEAST, POPNORTH))
```

```
spplot(map.resids, cuts=quantile(resids), col.regions=colours,
cex=0.3)
```

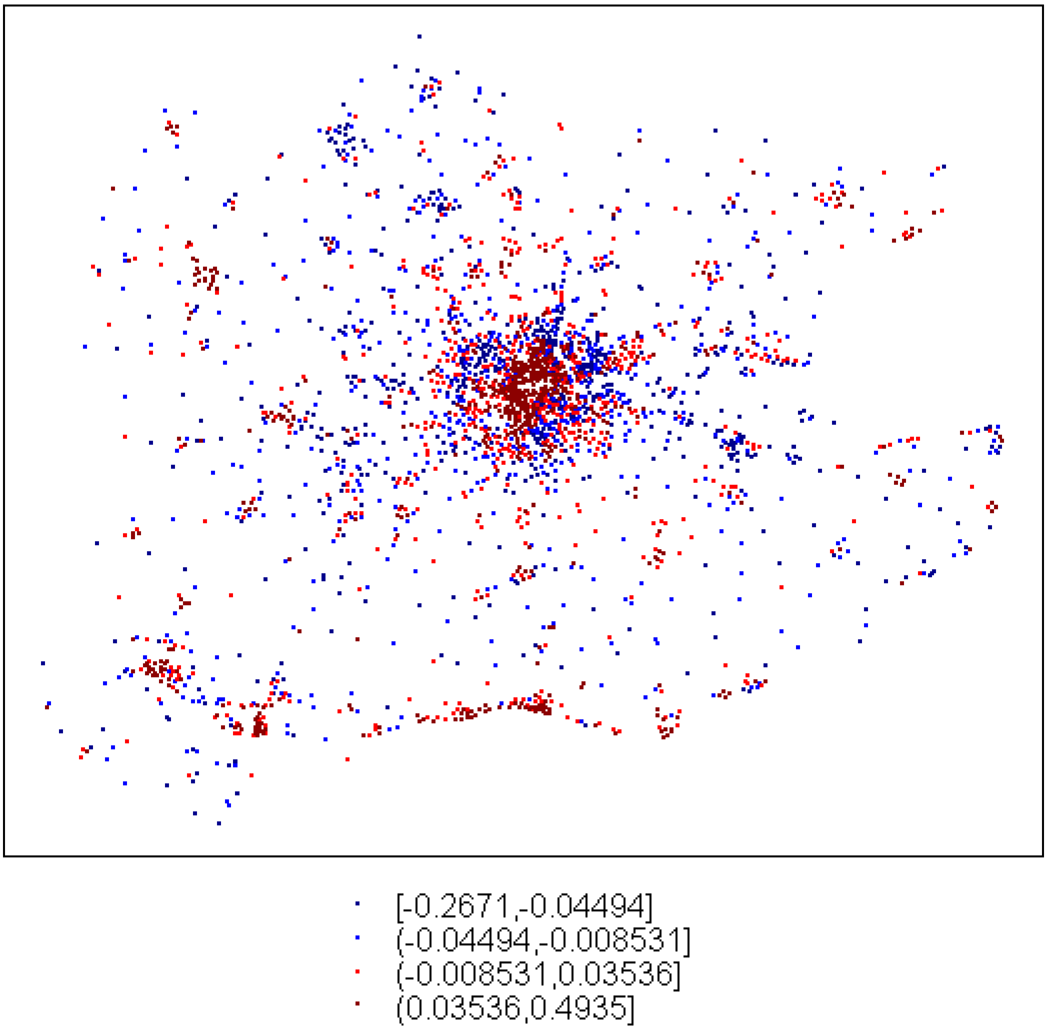


Figure 3. Map showing the geographical patterning of the regression residuals.

There does appear to be a pattern, invalidating an underlying assumption of the model: that the residuals have values independent of each other. There appears to be a spatial dependency that GWR might reveal further.

Geographically Weighted Regression

The basic idea behind GWR is to explore how the relationship between a dependent variable (Y) and one or more independent variables (the Xs) might vary geographically. Instead of assuming that a single model can be fitted to the entire study region, it looks for geographical differences.

GWR works by moving a search window from one point in a data set to the next, working through them all in sequence. As the search window rests on a sample point, all other points that are around it and within the search window are identified. A regression model is then fitted to that subset of the data, giving most weight to the points that are closest to the one at the centre. For a data set of 2536 observations GWR will, then, fit 2536 weighted regression models, the results of which are compared to look for geographical variation.

This immediately raises a question – what area should the search window cover each time? The answer is provided by a process of calibration, to select an "optimal" bandwidth (an optimal search window size).

Note that the distance from one point to another can be defined in two ways: either by actual geographic distance or by whether it's the first nearest neighbour, the second, the third and so forth. If the number of neighbours within the search window is fixed then it will vary in area from point to point: where the sample points are close together the window will have less area; where the points are sparse it will fill a greater area. This is called an adaptive window and is usually better for analysing census data (because census zones are of a variable size: smaller where population density is higher and *vice versa*).

To undertake GWR in R, first load the add-in:

```
library(spgwr)
```

Then, to calibrate the bandwidth, type the following. Be patient as it searches for the answer – it can take some time!

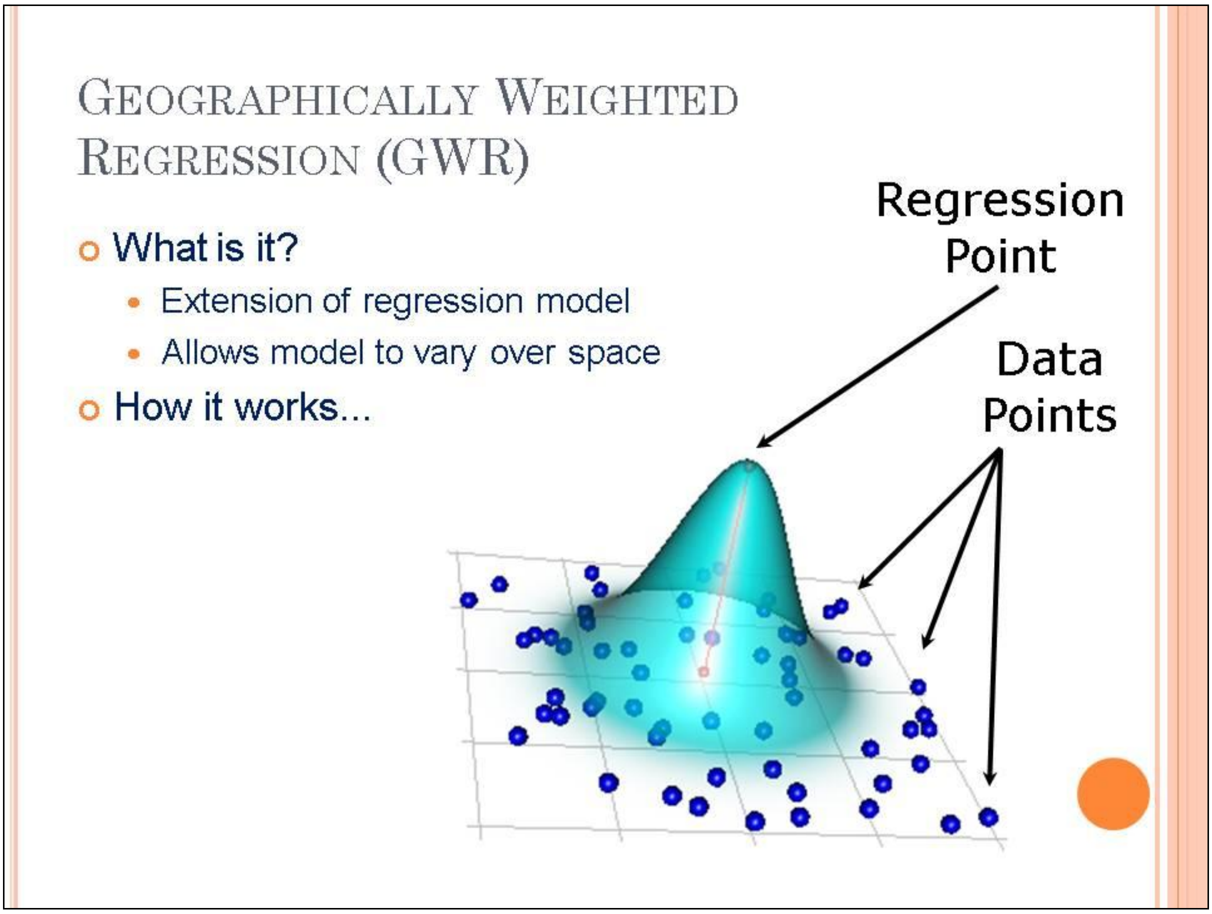
```
bw = gwr.sel(Nocars ~ Unemp, data=map, adapt=T)
```

To now fit the model, type,

```
gwr.model = gwr(Nocars ~ Unemp, data=map, adapt=bw)
```

And to see a summary of it

```
gwr.model
```



GEOGRAPHICALLY WEIGHTED REGRESSION (GWR)

- What is it?
  - Extension of regression model
  - Allows model to vary over space
- How it works…

Figure 4. GWR works by passing a search window from one point to the next and fitting a distance weighted regression model each time.
(Source: Prof. C. Brunsdon, University of Leicester)

What the summary is telling you is that in some places the relationship between car non ownership and unemployment is negative, whilst in others it is positive. Going across the 2536 sample points within the study region, for half of them, as unemployment rises by 1 percentage point then, all things being equal, the level of car non ownership will rise between 5.887 and 9.179 percentage points (these are the values marked as 1st Qu. and 3rd Qu. on your screen). However, if all the sample points are considered, the effect ranges from a decrease of 3.178 percentage points, to an increase of 25.580 percentage points.

We can anticipate a geographical patterning to the above, and there is!

```
spplot(gwr.model$SDF, "Unemp", cuts=quantile(gwr.model$SDF$Unemp),
col.regions=colours, cex=0.3)
```

```
# SDF is short for Spatial Data Frame and it is where the model
# results are stored
```

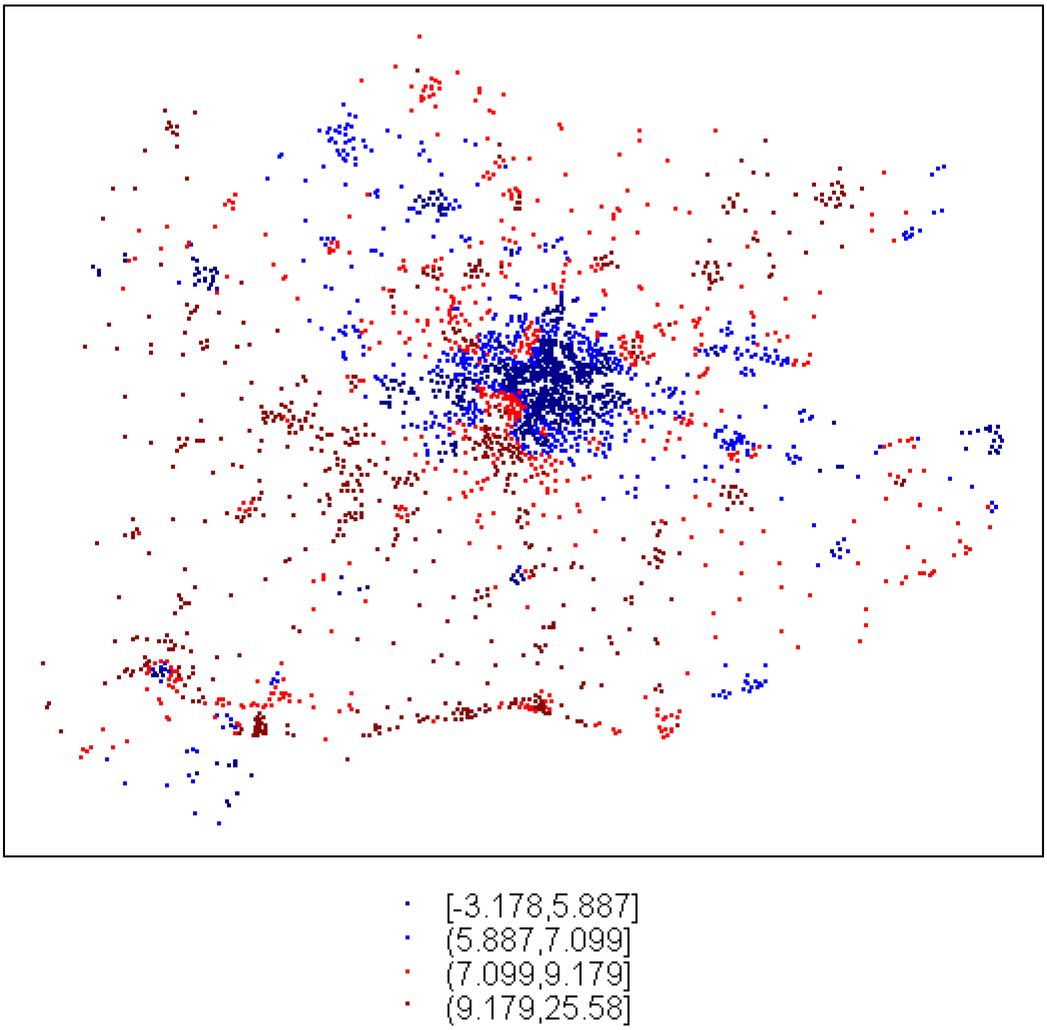


Figure 5. Mapping the results of the GWR model. The effect of unemployment on car non ownership appears to vary geographically.

What next?

We need to be careful not to assume that just because the relationship between car non ownership and unemployment varies geographically that is also a significant relationship everywhere. In some or more places it could be due to chance.

The following is for provided for information and it is recommended you don't try it as the fitting of the GWR model will take some time – about 10 or 15 minutes. What the code does is change the default settings to calculate measures of statistical significance and to plot the results. The resulting map is shown in Figure 6; in most places the relationship appears to be significant.

```
gwr.model = gwr(Nocars ~ Unemp, data=map, adapt=bw, hatmatrix=T,
se.fit=T)
```

```
t = gwr.model$SDF$Unemp / gwr.model$SDF$Unemp_se
```

```
sig.map = SpatialPointsDataFrame(map, data.frame(t))

colours=c("green","red","green")

breaks=c(min(t),-4,4,max(t))

spplot(sig.map, cuts=breaks, col.regions=colours, cex=c(0.3,1,0.3))
```

We also ought to think carefully about the form of our model. Does unemployment “cause” car non ownership or is it the other way around?

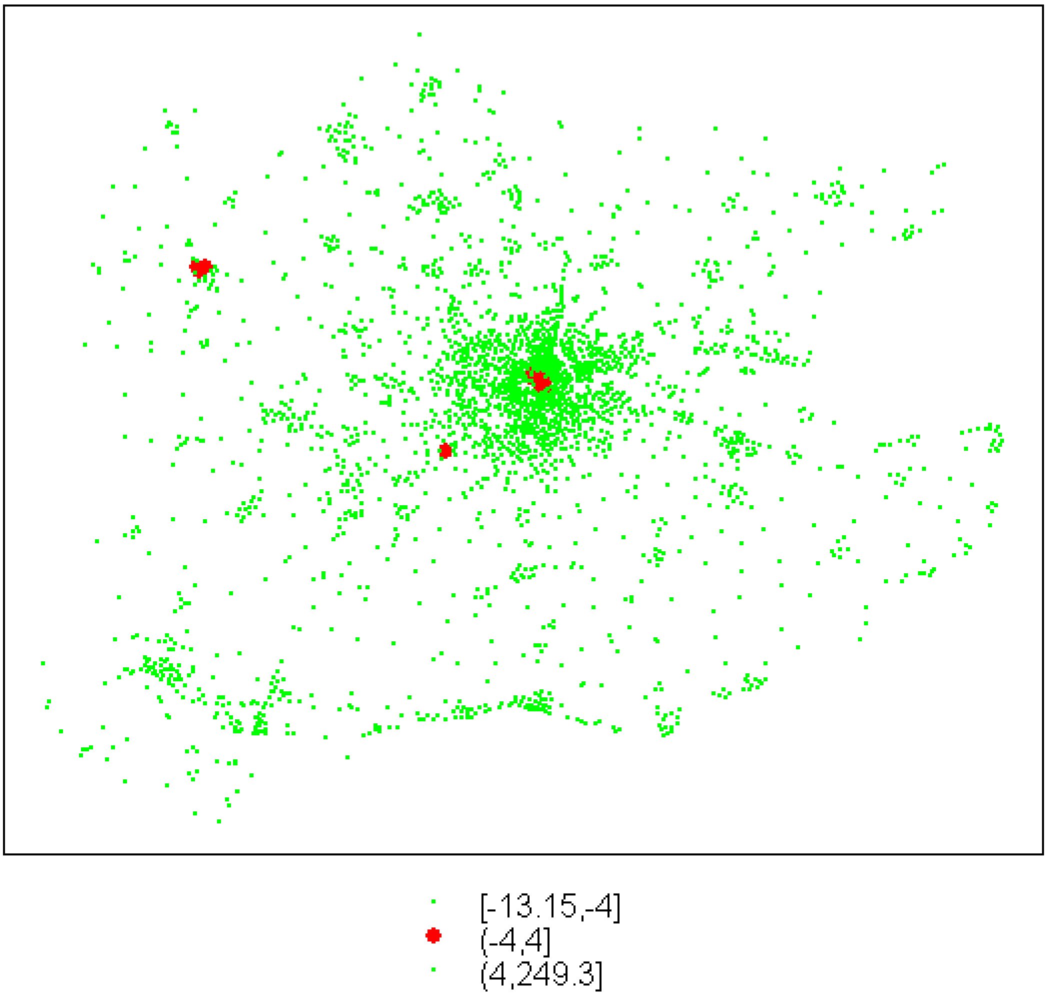


Figure 6. Estimated t values for the GWR model. Points shown in red indicate a relationship that is not significant.

Changing the model

We could now fit further models, using multiple regression to bring in more predictor variables to help explain patterns of car non ownership.

Note that if we do, or if we make other changes to the model, then the bandwidth needs recalibrating. For example:

```
bw2 = gwr.sel(Nocars ~ Unemp + DepChild, data=map, adapt=T)

gwr.model2 = gwr(Nocars ~ Unemp + DepChild, data=map, adapt=bw2)

gwr.model2
```

(etc.)

Further information

This has been a “whistle stop” tour through the basics of GWR. As stated at the beginning, the definitive text to consult for further information is Fotheringham *et al.* (2002) Geographically Weighted Regression: The Analysis of Spatially Varying Relationship, published by Wiley.

Note that the use of GWR has been made possible as part of an ongoing project for applied spatial analysis with R. For further details, read the book! Bivand et al. (2008) Applied Spatial Data Analysis with R, published by Springer. However, for a simpler and succinct introduction to R, read the free manual “An Introduction to R” at CRAN: http://cran.r-project.org/manuals.html

Finally, because of the way GWR works it take a *long* time to analyse larger data sets. For n = 150 000 observations about 2 weeks is required! Fortunately, methods are being developed to speed up the process using grid computing. Type “Grid enabled spatial regression models” or “Grid enabled GWR” into a suitable search engine to find out more.

RH, Friday, 06 February 2009